Problem A. Наибольшее паросочетание

Input file:	`стандартный ввод`
Output file:	`стандартный вывод`
Time limit:	0.5 секунды
Memory limit:	256 мебибайт

Ваша задача — найти наибольшее паросочетание в заданном двудольном графе.

Input

Первая строка входа содержит два целых числа n и m ($1 \le n, m \le 250$) — количество вершин в доле A и количество вершин в доле B, соответственно.

В последующих n строках заданы описания рёбер. Рёбра, выходящие из i-й вершины в A, заданы в $i+1$-й строке. Соответствующая строка содержит вершины из B, с которыми соединена соответствующая вершина в A. Вершины в обеих долях пронумерованы независимо, начиная с 1. Каждая строка завершается нулём.

Output

В первой строке выведите целое число l — количество рёбер в наибольшем паросочетании. Каждая из последующих l строк должна содержать два целых числа u_j, v_j и описывать одно ребро наибольшего паросочетания.

Example

стандартный ввод	стандартный вывод
2 2 1 2 0 2 0	2 1 1 2 2

Problem B. Минимизация наиболее дорогостоящего ребра

Input file:	`стандартный ввод`
Output file:	`стандартный вывод`
Time limit:	6 секунд
Memory limit:	256 мебибайт

Вам дан полный двудольный граф. Каждая доля содержит n вершин. Для каждого ребра (i, j) $(i, j = 1 \cdots n)$ задана его стоимость $g_{i,j}$.

Найдите такое совершенное паросочетание, в котором цена наиболее дорогостоящего ребра минимальна.

Input

Первая строка входа содержит одно целое число n $(1 \le n \le 500)$.

Далее задана матрица со стоимостями рёбер: каждая из последующих n строк содержит по n целых чисел; j-е число в i-й строке равно стоимости $g_{i,j}$ ребра, ведущего из вершины i из первой доли в вершину j во второй доле. Все $g_{i,j}$ — целые числа, по абсолютной величине не превосходящие 10^6.

Output

В первой строке выведите наименьшую возможную стоимость самого дорогостоящего ребра.

В последующих n строках выведите соответствующее совершенное паросочетание. Каждая из n строк должна содержать описание очередного ребра — номера соединяемых им вершин a_i и b_i.

Example

стандартный ввод	стандартный вывод
2 1 3 4 5	4 2 1 1 2

Problem C. Минимальное вершинное покрытие

Input file:	стандартный ввод
Output file:	стандартный вывод
Time limit:	1 секунда
Memory limit:	256 мебибайт

Вам дан граф и наибольшее паросочетание в нём. Ваша задача — найти минимальное вершинное покрытие в этом графе.

Input

Первая строка входа содержит два целых числа m и n ($1 \le m, n \le 4\,000$) — размеры долей графа.

i-я из последующих m строк содержит список рёбер, которые начинаются в i-й вершине первой доли. Список начинается с целого числа K_i ($0 \le K_i \le n$) — количества рёбер. После этого в произвольном порядке перечислены вершины из второй доли, соединённые с i-й вершиной первой доли. Гарантируется, что сумма всех K_i во входном файле не превосходит 500 000.

Последняя строка содержит m целых чисел $0 \le L_i \le n$ и задаёт наибольшее паросочетание; i-е число задаёт номер вершины из второй доли, соответствующий i-й вершине из первой доли, или равно 0, если i-й вершине из первой доли не сопоставлена ни одна вершина из второй доли.

Output

В первой строке выведите одно целое число — размер минимального вершинного покрытия.

Во второй строке сначала выведите целое число S — количество вершин в первой доле, принадлежащих минимальному вершинному покрытию, после чего выведите S целых чисел — номера вершин из первой доли, принадлежащих минимальному вершинному покрытию, отсортированные по возрастанию.

В третьей строке выведите в аналогичном формате список вершин из второй доли, принадлежащих минимальному вершинному покрытию.

Example

стандартный ввод	стандартный вывод
3 2 2 1 2 1 2 1 2 1 2 0	2 1 1 1 2

Problem D. Минимальное рёберное покрытие

Input file:	`стандартный ввод`
Output file:	`стандартный вывод`
Time limit:	0.5 секунды
Memory limit:	256 мебибайт

Дан двудольный граф. Ваша задача — найти его минимальное рёберное покрытие.

Набор рёбер R называется *рёберным покрытием* тогда и только тогда, когда каждая вершина заданного графа принадлежит хотя бы одному ребру из R.

Изолированные вершины допустимы и при этом не должны быть покрыты, то есть в этой задаче рёберным покрытием является такое подмножество рёбер R, что каждая неизолированная вершина покрыта.

Input

Первая строка входа содержит два целых числа n и m ($1 \le n, m \le 250$) — количество вершин в доле A и количество вершин в доле B соответственно.

Последующие n строк задают граф. i-я вершина из доли A описана в $i + 1$-й строке, которая содержит список номеров вершин из доли B, соединённых с сооответствующей вершиной из доли A и терминированный нулём. Вершины в каждой доле пронумерованы независимо.

Output

Первая строка должна содержать целое число l — количество рёбер в минимальном рёберном покрытии. Каждая из последующих l строк должна содержать два целых числа u_j и v_j — вершины, соединённых очередным ребром из минимального рёберного покрытия. Если решений несколько, выведите любое.

Напоминаем, что если в графе есть изолированные вершины, они не должны входить в рёберное покрытие.

Example

стандартный ввод	стандартный вывод
2 2 1 2 0 2 0	2 1 1 2 2

Problem E. Minimum Path Cover

Input file:	стандартный ввод
Output file:	стандартный вывод
Time limit:	0.5 секунды
Memory limit:	256 мебибайт

Дан ориентированный ациклический граф. Требуется найти такой минимальный набор путей, что каждая из вершин принадлежит ровно одному пути. Заметим, что пути могут иметь нулевую длину (то есть состоять ровно из одной вершины).

Input

Первая строка содержит два целых числа n и m — количество вершин и количество рёбер в графе ($1 \le n \le 100$).

Каждая из последующих m строк содержит два целых числа — номер вершины, из которой идёт ребро, и номер вершины, в которую идёт ребро.

Гарантируется, что граф ациклический, а также что граф не содержит петель и кратных рёбер.

Output

В первой строке выведите одно целое число — минимальное количество путей. Далее выведите пути по одному на строку (формат см. в примере к задаче). Если решений несколько, выведите любое.

Examples

стандартный ввод	стандартный вывод
4 4 1 2 2 4 1 3 3 4	2 1 2 4 3
7 8 1 2 1 3 2 4 3 4 4 5 4 6 5 7 6 7	3 1 2 4 5 7 3 6